CSCI 5832
Natural Language Processing

Jim Martin
Lecture 15

Today 3/6

- Full Parsing
  - Review Earley
- Partial Parsing & Chunking
  - FST/Cascades
  - Sequence classification

Earley Example

- Book that flight
- We should find… an S from 0 to 3 that is a completed state…

Example

Chart[0]	S0	$\gamma \rightarrow \bullet S$	[0,0]	Dummy start state
	S1	$S \rightarrow \bullet NP\ VP$	[0,0]	Predictor
	S2	$S \rightarrow \bullet Aux\ NP\ VP$	[0,0]	Predictor
	S3	$S \rightarrow \bullet VP$	[0,0]	Predictor
	S4	$NP \rightarrow \bullet Pronoun$	[0,0]	Predictor
	S5	$NP \rightarrow \bullet Proper\text{-}Noun$	[0,0]	Predictor
	S6	$NP \rightarrow \bullet Det\ Nominal$	[0,0]	Predictor
	S7	$VP \rightarrow \bullet Verb$	[0,0]	Predictor
	S8	$VP \rightarrow \bullet Verb\ NP$	[0,0]	Predictor
	S9	$VP \rightarrow \bullet Verb\ NP\ PP$	[0,0]	Predictor
	S10	$VP \rightarrow \bullet Verb\ PP$	[0,0]	Predictor
	S11	$VP \rightarrow \bullet VP\ PP$	[0,0]	Predictor

Example

Chart[1]	S12	$Verb \rightarrow book \bullet$	[0,1]	Scanner
	S13	$VP \rightarrow Verb \bullet$	[0,1]	Completer
	S14	$VP \rightarrow Verb \bullet NP$	[0,1]	Completer
	S15	$VP \rightarrow Verb \bullet NP\ PP$	[0,1]	Completer
	S16	$VP \rightarrow Verb \bullet PP$	[0,1]	Completer
	S17	$S \rightarrow VP \bullet$	[0,1]	Completer
	S18	$VP \rightarrow VP \bullet PP$	[0,1]	Completer
	S19	$NP \rightarrow \bullet Pronoun$	[1,1]	Predictor
	S20	$NP \rightarrow \bullet Proper\text{-}Noun$	[1,1]	Predictor
	S21	$NP \rightarrow \bullet Det\ Nominal$	[1,1]	Predictor
	S22	$PP \rightarrow \bullet Prep\ NP$	[1,1]	Predictor

Example

Chart[2]	S23	$Det \rightarrow that \bullet$	[1,2]	Scanner
	S24	$NP \rightarrow Det \bullet Nominal$	[1,2]	Completer
	S25	$Nominal \rightarrow \bullet Noun$	[2,2]	Predictor
	S26	$Nominal \rightarrow \bullet Nominal\ Noun$	[2,2]	Predictor
	S27	$Nominal \rightarrow \bullet Nominal\ PP$	[2,2]	Predictor
Chart[3]	S28	$Noun \rightarrow flight \bullet$	[2,3]	Scanner
	S29	$Nominal \rightarrow Noun \bullet$	[2,3]	Completer
	S30	$NP \rightarrow Det\ Nominal \bullet$	[1,3]	Completer
	S31	$Nominal \rightarrow Nominal \bullet Noun$	[2,3]	Completer
	S32	$Nominal \rightarrow Nominal \bullet PP$	[2,3]	Completer
	S33	$VP \rightarrow Verb\ NP \bullet$	[0,3]	Completer
	S34	$VP \rightarrow Verb\ NP \bullet PP$	[0,3]	Completer
	S35	$PP \rightarrow \bullet Prep\ NP$	[3,3]	Predictor
	S36	$S \rightarrow VP \bullet$	[0,3]	Completer
	S37	$VP \rightarrow VP \bullet PP$	[0,3]	Completer

Efficiency

- For such a simple example, there seems to be a lot of useless stuff in there.
- Why?
  - It's predicting things that aren't consistent with the input
  - That's the flipside to the CKY problem.

3/11/08 7

Details

- As with CKY that isn't a parser until we add the backpointers so that each state knows where it came from.

3/11/08 8

Full Syntactic Parsing

- Probably necessary for deep semantic analysis of texts (as we'll see).
- Probably not practical for many applications (given typical resources)
  - O(n^3) for straight parsing
  - O(n^5) for probabilistic versions
  - Too slow for applications that need to process texts in real time (search engines)
  - Or that need to deal with large volumes of new material over short periods of time

3/11/08 9

Partial Parsing

- For many applications you don't really need a full-blown syntactic parse. You just need a good idea of where the base syntactic units are.
  - Often referred to as chunks.
- For example, if you're interested in locating all the people, places and organizations in a text it might be useful to know where all the NPs are.

3/11/08 10

Examples

[$_{NP}$ The morning flight] [$_{PP}$ from] [$_{NP}$ Denver] [$_{VP}$ has arrived.]

[$_{NP}$ a flight] [$_{PP}$ from] [$_{NP}$ Indianapolis][$_{PP}$ to][$_{NP}$ Houston][$_{PP}$ on][$_{NP}$ TWA]

[$_{NP}$ The morning flight] from [$_{NP}$ Denver] has arrived.

- The first two are examples of full partial parsing or chunking. All of the elements in the text are part of a chunk. And the chunks are non-overlapping.
- Note how the second example has no hierarchical structure.
- The last example illustrates base-NP chunking. Ignore anything that isn't in the kind of chunk you're looking for.

3/11/08 11

Partial Parsing

- Two approaches
  - Rule-based (hierarchical) transduction.
  - Statistical sequence labeling
    - HMMs
    - MEMMs

3/11/08 12

Rule-Based Partial Parsing

- Restrict the form of rules to exclude recursion (make the rules flat).
- Group and order the rules so that the RHS of the rules can refer to non-terminals introduced in earlier transducers, but not later ones.
- Combine the rules in a group in the same way we did with the rules for spelling changes.
- Combine the groups into a cascade…
- Then compose, determinize and minimize the whole thing (optional).

3/11/08 13

Typical Architecture

- Phase 1: Part of speech tags
- Phase 2: Base syntactic phrases
- Phase 3: Larger verb and noun groups
- Phase 4: Sentential level rules

3/11/08 14

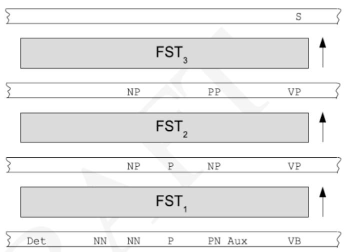
Partial Parsing

$$NP \rightarrow (Det)\ Noun^{*}\ Noun$$
$$NP \rightarrow Proper\text{-}Noun$$
$$VP \rightarrow Verb$$
$$VP \rightarrow Aux\ Verb$$

- No direct or indirect recursion allowed in these rules.
- That is you can't directly or indirectly reference the LHS of the rule on the RHS.

3/11/08 15

Cascaded Transducers



The morning flight from Denver has arrived

Partial Parsing

- This cascaded approach can be used to find the sequence of flat chunks you're interested in.
- Or it can be used to approximate the kind of hierarchical trees you get from full parsing with a CFG.

Break

- Quiz is on 3/18. It will cover
  - 12, 13 and 14 and relevant parts of 6
  - Same format as last time except that...
    - You can bring a 1 page cheat sheet (1 side)
    - 1 page on which you can write anything you think might be helpful

Statistical Sequence Labeling

- As with POS tagging, we can use rules to do partial parsing or we can train systems to do it for us. To do that we need training data and the right kind of encoding.
  - Training data
    - Hand tag a bunch of data (as with POS tagging)
    - Or even better, extract partial parse bracketing information from a treebank.

Encoding

- With the right encoding you can turn the labeled bracketing task into a tagging task. And then proceed exactly as we did with POS Tagging.
- We'll use what's called IOB labeling to do this.
  - I -> Inside
  - O -> Outside
  - B -> Begin

IOB encoding

The	*morning*	*flight*	*from*	*Denver*	*has*	*arrived.*
B_NP	I_NP	I_NP	O	B_NP	O	O

- This first example shows the encoding for just base-NPs. There are 3 tags in this scheme.

The	*morning*	*flight*	*from*	*Denver*	*has*	*arrived*
B_NP	I_NP	I_NP	B_PP	B_NP	B_VP	I_VP

- This example shows full coverage. In this scheme there are 2*N+1 tags. Where N is the number of constituents in your set.

Methods

- HMMs
- Sequence Classification
  - Using any kind of standard ML-based classifier.

3/11/08

22

Evaluation

- Suppose you employ this scheme. What's the best way to measure performance.
- Probably not the per-tag accuracy we used for POS tagging.
  - Why?
    - It's not measuring what we care about
    - We need a metric that looks at the chunks not the tags

3/11/08

23

Example

- Suppose we were looking for PP chunks for some reason.
- If the system simple said O all the time it would do pretty well on a per-label basis since most words reside outside any PP.

3/11/08

24

Precision/Recall/F

- Precision:
  - The fraction of chunks the system returned that were right
    - "Right" means the boundaries and the label are correct given some labeled test set.
- Recall:
  - The fraction of the chunks that system got from those that it should have gotten.
- F: Harmonic mean of those two numbers.

3/11/08 25

HMM Tagging

- Same as with POS tagging
  - Argmax P(T|W) = P(W|T)P(T)
  - The tags are the hidden states
- Works ok but it isn't great.
  - The typical kinds of things that we might think would be useful in this task aren't easily squeezed into the HMM model
- We'd like to be able to make arbitrary features available for the statistical inference being made.

3/11/08 26

Supervised Classification

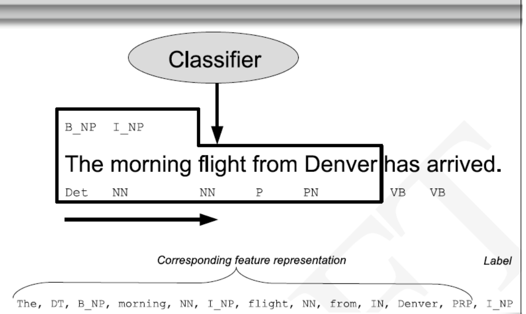
- Training a system to take an object represented as a set of features and apply a label to that object.
- Methods typically include
  - Naïve Bayes
  - Decision Trees
  - Maximum Entropy (logistic regression)
  - Support Vector Machines
  - …

3/11/08 27

Sequence Classification

- Applying this to tagging…
  - The object to be tagged is a word in the sequence
  - The features are
    - features of the word,
    - features of its immediate neighbors,
    - and features derived from the entire sentence.
  - Sequential tagging means sweeping the classifier across the input assigning tags to words as you proceed.

Statistical Sequence Labeling



Typical Features

- Typical setup involves
  - A small sliding window around the object being tagged
  - Features extracted from the window
    - Current word token
    - Previous/next N word tokens
    - Current word POS
    - Previous/next POS
    - Previous N chunk labels
    - Capitalization information
    - ...

Performance

- With a decent ML classifier
  - SVMs
  - Maxent
  - Even decision trees
- You can get decent performance with this arrangement.
- Good CONLL 2000 scores had F-measures in the mid-90s.

3/11/08 31

Problem

- You're making a long series of local judgments. Without attending to the overall goodness of the final sequence of tags. You're just hoping that local conditions will yield global goodness.
- Note that HMMs didn't have this problem since the language model worried about the overall goodness of the tag sequence.
  - But we don't want to use HMMs since we can't easily squeeze arbitrary features into the

3/11/08 32

Answer

- Graft a language model onto the sequential classification scheme.
  - Instead of having the classifier emit one label as an answer for each object, get it to emit an N-best list for each judgment.
  - Train a language model for the kinds of sequences we're trying to produce.
  - Run Viterbi over the N-best lists for the sequence to get the best overall sequence.

3/11/08 33

MEMMs

- Maximum entropy Markov models are the current standard way of doing this.
  - Although people do the same thing in an ad hoc way with SVMs.
- MEMMs combine two techniques
  - Maximum entropy (logistic) classifiers for the individual labeling
  - Markov models for the sequence model.

Models

- HMMs and graphical models are often referred to as generative models since they're based on using Bayes…
  - So to get P(c|x) we use P(x|c)P(c)
- Alternatively we could use what are called discriminative models; models that get P(c|x) directly without the Bayesian inversion

MaxEnt

- Multinomial logistic regression
- Along with SVMs, Maxent is the typical technique used in NLP these days when a classifier is required.
  - Provides a probability distribution over the classes of interest
  - Admits a wide variety of features
  - Permits the hand-creation of complex features
  - Training time isn't bad

MaxEnt

$$p(c|x) = \frac{1}{Z} \exp \sum_i w_i f_i$$

3/11/08

37

MaxEnt

$$p(c|x) = \frac{\exp\left(\sum_{i=0}^{N} w_{ci} f_i\right)}{\sum_{c' \in C} \exp\left(\sum_{i=0}^{N} w_{c'i} f_i\right)}$$

3/11/08

38

Hard Classification

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|x)$$

- If we really want an answer…
- But typically we want a distribution over the answers.

3/11/08

39

MaxEnt Features

- They're a little different from the typical supervised ML approach
  - Limited to binary values
    - Think of a feature as being on or off rather than as a feature with a value
  - Feature values are relative to an object/class pair rather than being a function of the object alone.
  - Typically have lots and lots of features (100,000s of features isn't unusual.)

3/11/08 40

Features

$$f_3(c,x) = \begin{cases} 1 & \text{if suffix}(word_i) = \text{“ing”} \ \& \ c = \text{VBG} \\ 0 & \text{otherwise} \end{cases}$$

$$f_4(c,x) = \begin{cases} 1 & \text{if is\_lower\_case}(word_i) \ \& \ c = \text{VB} \\ 0 & \text{otherwise} \end{cases}$$

3/11/08 41

Features

$$f_3(c,x) = \begin{cases} 1 & \text{if suffix}(word_i) = \text{“ing”} \ \& \ c = \text{VBG} \\ 0 & \text{otherwise} \end{cases}$$

$$f_4(c,x) = \begin{cases} 1 & \text{if is\_lower\_case}(word_i) \ \& \ c = \text{VB} \\ 0 & \text{otherwise} \end{cases}$$

- Key point. You can't squeeze features like these into an HMM.

3/11/08 42

Mega Features

$$f_{125}(c,x) = \begin{cases} 1 & \text{if } word_{i-1} = \texttt{<s>} \ \& \ \text{isupperfirst}(word_i) \ \& \ c = \text{NNP} \\ 0 & \text{otherwise} \end{cases}$$

- These have to be hand-crafted.
- With the right kind of kernel they can be exploited implicitly with SVMs. At the cost of a increase in training time.

3/11/08 43

Back to Sequences

$$\hat{T} = \underset{T}{\operatorname{argmax}} P(T|W)$$
$$= \underset{T}{\operatorname{argmax}} P(W|T)P(T)$$
$$= \underset{T}{\operatorname{argmax}} \prod_i P(word_i|tag_i) \prod_i P(tag_i|tag_{i-1})$$

- HMMs

$$\hat{T} = \underset{T}{\operatorname{argmax}} P(T|W)$$
$$= \underset{T}{\operatorname{argmax}} \prod_i P(tag_i|word_i, tag_{i-1})$$

- MEMMs

And whatever other features you choose to use!

3/11/08 44

Back to Viterbi

$$v_t(j) = \max_{1 \leq i \leq N-1} v_{t-1}(i)\, P(s_j|s_i, o_t); \quad 1 < j < N, 1 < t < T$$

- The value for a cell is found by examining all the cells in the previous column and multiplying by the posterior for the current column (which incorporates the transition as a factor, along with any other features you like).

3/11/08 45

Next Time

- Statistical Parsing (Chapter 14)

3/11/08 46